7 The Sound Manager.

The Sound Manager deals with the sound chip. It allows various envelopes and sounds to be set up and played under the control of the user. Most of the control is achieved using software rather than the sound chip hardware.

7.1 The Sound Chip.

The sound chip used is the General Instruments AY-3-8912. This has three channels and a pseudo-random noise generator that can be connected to any of the channels. The chip has a limited number of amplitude envelopes available (see Appendix IX) but the software enveloping, described below, can achieve all that the hardware is capable of, and more. Tone enveloping is all done by the software, there is no hardware support.

The sound generated by the chip uses square waveforms. There is no way to generate any other waveform.

It is possible to access the sound chip directly should the need arise. However, the routine MC SOUND REGISTER should be used to write to registers of the sound chip. This is because the keyboard is attached to the 1/0 port of the sound chip and the keyboard scanning routine expects to find the sound chip in a standard state (i.e. not in use). Also, there are timing constraints on accesses to the chip; using MC SOUND REGISTER will avoid consideration of these details.

The sound chip has three independent sound channels. The outputs from these are mixed together to form two stereo channels - sound channels A and B are mixed to form one stereo channel and sound channels B and C are mixed to form the other stereo channel. The stereo sound is available on the output jack on the back of the machine. However, there is only a single internal speaker and so the two stereo channels are mixed together to drive this. The volume of sound from the internal speaker can be controlled by the volume control knob on the side of the machine near the on/off switch. This control overrides the other volume control methods described below.

7.2 Tone Periods and Amplitudes.

The sound chip allows 16 different amplitudes in the range 0..15. Amplitude 0 is no sound at all, amplitude 15 is maximum volume.

The pitch of a note to be generated is specified by the period of the note rather than by the frequency. This period is given in 8 microsecond units. Thus, the tone period specified and the frequency of the tone generated are related by the formula:

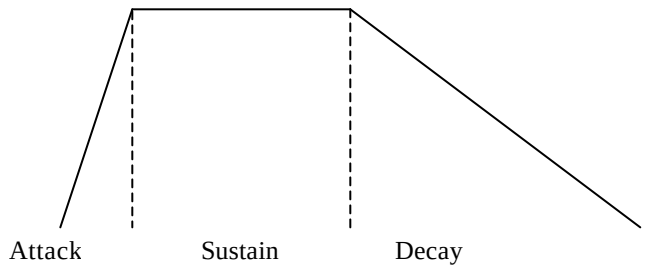
Tone period = 125 000 / Frequency

See Appendix VIII for a list of the suggested periods for generating musical notes.

7.3 Enveloping.

Real sounds rarely have a constant volume. Enveloping allows an approximation to the variation in volume of real sounds to be made. The sound is split into a number of sections each of which can increase the volume, decrease the volume, or keep it constant. The length of these sections can be varied, as can the rate of increase or decrease in volume. For example, a note generated by a musical instrument may be considered to have 3 sections as follows:

Attack:	The volume of the note rises rapidly to its peak.
Sustain:	The volume of the note remains constant while the note is played.
Decay:	The volume falls away slowly to zero as the note finishes.



The Sound Manager allows two types of envelopes; amplitude envelopes to control a sound's volume and tone envelopes to control its pitch (the pitch is varied in much the same way as the volume). The user can set up to 15 different envelopes of each type. The exact formats of the data blocks specifying envelopes are given in SOUND AMPL ENVELOPE and SOUND TONE ENVELOPE.

a. Amplitude envelopes.

An amplitude envelope is used to control the volume and length of a sound. It can have up to five sections. Each section can be either a hardware or a software section. Software sections are either absolute or relative.

Hardware sections write values into the sound chip registers 11, 12 and 13 to set up a hardware envelope. (See Appendix IX for a description of the sound chip registers). Generally a hardware section will be followed by a software section that does nothing except wait for a time long enough for the hardware envelope to operate.

An absolute software section specifies a volume to set and a time to wait before obeying the next section.

A relative software section specifies an step size, a number of steps and a time to wait. For each step requested, the current volume is changed by the given step size and then the Sound Manager waits for the given time after each step before obeying the next step.

Amplitude envelopes are set by calling SOUND AMPL ENVELOPE.

b. Tone envelopes.

A tone envelope controls the pitch of the sound. It can have up to five sections. Each section can be either an absolute or a relative section. The sections of a tone envelope are not necessarily related to those of an amplitude envelope.

An absolute section specifies a tone period to set and a time to wait before obeying the next section.

A relative section specifies an step size, a number of steps and a time to wait. For each step requested, the current tone period is changed by the given step size and then the Sound Manager waits for the given time after each step before obeying the next step.

If the tone envelope is completed before the sound duration expires (see section 7.4f) then the final pitch is held constant. Alternatively, tone envelopes can be set to repeat themselves automatically. This allows tremulo effects to be created.

Tone envelopes are set by calling SOUND TONE ENVELOPE.

7.4 Sound Commands.

When a sound is given to the Sound Manager to be played, by calling SOUND QUEUE, a lot of information needs to be specified. This is described briefly below. The detailed layout of a sound command data block is described in SOUND QUEUE.

a. Initial tone period.

The sound is issued with an initial tone period. The pitch of the sound can be varied from this initial value using a tone envelope. If no tone envelope is specified the pitch remains constant. An initial tone period of zero means no tone is to be generated, presumably the sound is to be pure noise (see (e) below).

b. Initial volume.

The sound is issued with an initial volume. The volume of the sound can be varied from this initial value using an amplitude envelope. If no amplitude envelope is specified then the volume remains constant.

c. Tone envelope.

This specifies which tone envelope to use. If no envelope is specified then the pitch of the sound remains constant.

d. Amplitude envelope.

This specifies which amplitude envelope to use. If no envelope is specified then default system envelope is used. This keeps the volume of the sound constant and lasts for 2 seconds.

e. Noise period.

If the noise period is zero then no noise is to be added to the sound. Any other value sets the period for the pseudorandom noise generator and adds noise to the tone generated. Note that there is only one noise generator and so if two sounds are to use it at the same time they will need to agree on the period.

f. Duration.

The length of a sound can be specified in two ways, either as an absolute time (duration) or as a number of operations of the amplitude envelope. In the latter case the envelope is run one or more times and the sound finishes when the envelope has been executed the specified number of times. In the former case, if the duration finishes before the envelope (if any) then the sound is cut short. If the duration is longer than the envelope then the final amplitude is held until the duration expires.

g. Channels and Synchronisation Bits.

The sound can be issued to one or more channels. If a sound is issued to more than one channel then these channels automatically rendezvous with each other. Rendezvous requirements can be set explicitly as well. Also the sound can be held or the sound queue can be flushed (see section 7.6).

7.5 Sound Queues.

Each channel has a queue associated with it. Each queue has space to store at least three sounds. The sound at the head of each queue may be running and making music on its channel or it may be waiting for various synchronisation requirements (see 7.6 below). When a sound command is issued the sound is placed into the queues for the channels specified by the command. When the sound reaches the head of the queue, and providing its synchronisation requirements are met, it is executed.

If a sound that has the flush bit set is put into a queue then all sounds queued for that channel are discarded and any executing sound is stopped immediately. Thus a sound with the flush bit set will move to the head of the queue immediately and may commence execution.

A routine (SOUND CHECK) is provided to test the status of the sound at the head of a queue and to determine how much free space is in a queue. It is also possible to set up a sound event for each queue (by calling SOUND ARM EVENT). This synchronous event is 'kicked' when the queue has a free space in it. The sound event mechanism allows the generation of sound to be carried on as a background task whilst some other action is being carried out.

7.6 Synchronisation.

There are two mechanisms to allow sounds on different channels to be synchronised. These are holding sounds and rendezvous. The purpose of synchronisation is to ensure that sounds start simultaneously. For example, a simulation of an instrument might use one channel to generate the fundamental note and another channel to generate the harmonics of the note. The synchronisation mechanism, particularly rendezvous, may be used to ensure that the fundamental and the harmonic sounds start exactly together.

A sound can be specified to be held when it is issued. This means that when it reaches the head of the sound queue it is not executed immediately. Instead it waits until it is explicitly released (by calling SOUND RELEASE) before it starts execution.

A sound can have rendezvous requirements set on it when it is issued. If a sound is issued to more than one channel then these channels all set rendezvous with each other automatically. When a sound with a rendezvous set reaches the head of the sound queue it is not executed immediately. Instead it waits until sounds with matching rendezvous requirements reach the head of their sound queues. Only when all rendezvous sounds are found to be present and ready to run do they start.

For instance, a sound on channel A marked to rendezvous with a sound on channel B will not start until a sound on channel B marked to rendezvous with channel A is ready to start - and vice versa! If a sound is ready to start on channel B that is not marked to rendezvous with channel A then it starts but the sound on channel A continues to wait for its rendezvous.

7.7 Holding Sounds.

It is possible to stop a sound while it is executing by calling SOUND HOLD. This will stop a channel making any sound and will save the state of the sound. The sound can be restarted from where it was held by calling SOUND CONTINUE. However, if a hardware envelope was running when the sound was held then it is impossible to predict the effect of restarting the sound. The hardware envelope may or may not continue from where it was held.

Calling SOUND HOLD is different from setting the hold bit when issuing a sound as described in section 7.6 above. SOUND HOLD stops all sounds being generated at any time whilst the hold bit is a method for synchronising sounds and prevents a particular sound starting when it reaches the head of the queue.